**Maximizing Resource Utilization in Distributed Systems**

**Name**: Sai Kiran Nagarajan

**Student** **ID**: 45506043

**Introduction**

The goal of this project is to provide a new Java client that can dispatch/ schedule jobs to servers in a distributed system. In most cases, scheduling is constrained by factors such as execution time, turnaround time, resource utilisation, and execution cost. Our task is to come up with at least one novel scheduling method that can improve one of the aforementioned contraints. Our new scheduling algorithm should strive to reduce average turnaround time, lower total server costs, or increase average resource usage. We must keep in mind when constructing this new scheduling algorithm that optimising one of the criteria may result in the optimisation of the other constraints being sacrificed. As a result, we must be cautious when constructing the new algorithm and make appropriate concessions. We must also compare and contrast our algorithm outcomes with the three baseline methods.

**Problem Definition**

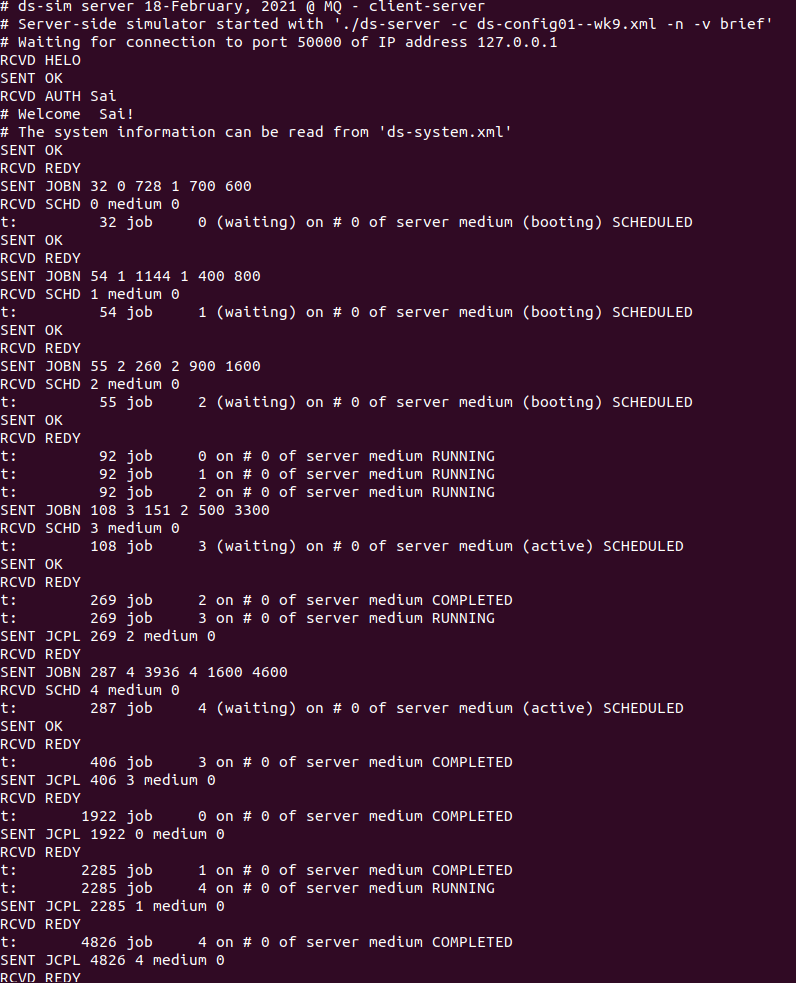
In this project, we have a difficulty in that when we despatch and/or schedule jobs to servers in a distributed system, they usually have a lot of constraints, such as execution time, turnaround time, resource use, and cost. We made effort to be cautious when building our new method because optimising restrictions can sometimes necessitate the sacrifice of another requirement. We optimised the average resource use in our newly devised method, albeit at the expense of increasing the turnaround time.

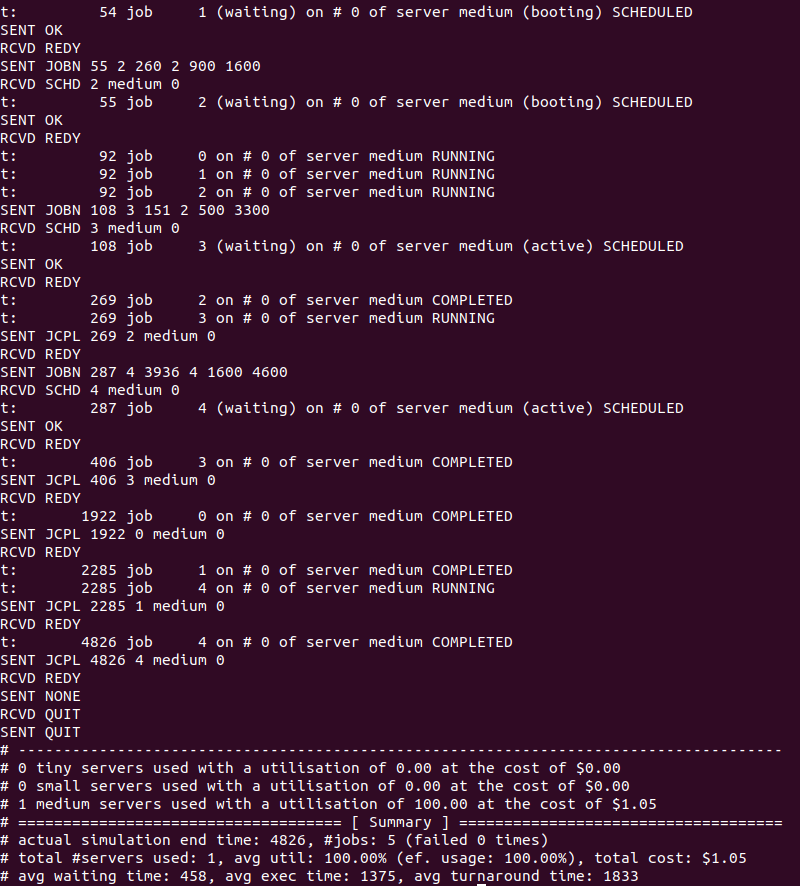
**Algorithm Description**

Sample Config:



We use all to largest algorithm by finding the largest server and then scheduling the job to that one.

Below are example scheduling scenarios.



**Implementation Details/ Data Structures**

I used Socket, DataInputStream and PrintStream from the java library.

Data structures used are string and integer arrays.

**Evaluation**

There are a number of reasons why we created our algorithm in this manner. This is because, when scheduling jobs in a distributed system, we aimed to maximise resource use. This improved one limitation but had the unintended consequence of affecting another. However, the disadvantages of our newly devised algorithm were that it maximised resource consumption while simultaneously increasing turnaround time.

**Conclusion**

Finally, we were asked to build and build a new Java client that can dispatch/ schedule jobs to servers in distributed systems for this project. When it comes to scheduling jobs to servers, there are a number of concerns and constraints to consider, including turnaround time, resource usage, execution time, and cost. So it was up to me to come up with a new scheduling method that could overcome at least one of these limits. My method was created to maximise resource use at the expense of increasing turnaround time. Overall, you can improve some of the limits but at the expense of others.

**Reference:**

https://www.javatpoint.com/socket-programming

http://www.cs.mun.ca/~harold/Courses/Old/CS1000.W19/Files/Textbook5th.pdf

<https://github.com/SaiKiran-26/DistributedSystems-Assignment2>